Question 1: Will the following program (eventually) terminate? Assume that reading or writing a single variable is atomic.

\[ i \leftarrow 0 \]
\[ j \leftarrow 0 \]

\[ \text{thread while } i = 0 \]
\[ \quad \text{DO } j \leftarrow j + 1 \mod 2 \; ; \text{print } i \]
\[ \quad \text{print } i \]

\[ \text{thread while } i = 0 \]
\[ \quad \text{do if } j = 0 \text{ then } i = 1 \]

Question 2: Will the following program (eventually) terminate, or is it possible that it runs forever? Assume that reading or writing a single variable is atomic.

\[ a \leftarrow 1 \]
\[ b \leftarrow 1 \]

\[ \text{thread while } a \neq 0 \]
\[ \quad \text{do } b \leftarrow (b + a) \mod 2 \]

\[ \text{thread while } b \neq 0 \]
\[ \quad \text{do } a \leftarrow a + 1 \]
\[ \quad a \leftarrow 0 \]

Question 3: Lamport’s logical clock algorithm works in the message passing model. Modify Lamport’s logical clock algorithm to assign logical clock values to all events in a shared memory system that supports atomic reads and atomic writes to shared memory. Prove that the logical clock created by your algorithm can be used to put the events in a total order \( \langle A, \Rightarrow \rangle \) consistent with the partial order \( \langle A, \rightarrow \rangle \).